****

**TRIBHUVAN UNIVERSITY**

**INSTITUTE OF ENGINEERING**

PULCHOWK CAMPUS

A REPORT ON

OOP in Python Language

SUBMITTED BY:

SUSHANT THAKUR (081BEL092)

SUBMITTED TO:

DEPARTMENT OF ELECTRONICS & COMPUTER ENGINEERING

**QUESTION 1:**

Create a class Book with attributes title, author, and price. Write a constructor to initialize these values and create an object with sample data.

● Add a method display\_info() to the Book class that prints the book's title, author, and price. Call this method using a Book object.

● Add a method update\_price(new\_price) to the Book class that updates the book's price. Demonstrate how to use it with an object.

**CODE 1:**

class Book:

def \_\_init\_\_(self, title, author, price):

self.title = title

self.author = author

self.price = price

def display\_info(self):

print(f"Title: {self.title}, Author: {self.author}, Price: {self.price}")

def update\_price(self, new\_price):

self.price = new\_price

print(f"Price updated to {self.price}")

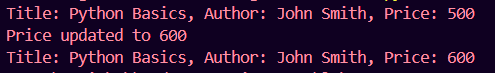
b1 = Book("Python Basics", "John Smith", 500)

b1.display\_info()

b1.update\_price(600)

b1.display\_info()

**OUTPUT 1:**



**OUESTION 2:**

Create a class Student with attributes name and marks. Create three objects of the class and display their details using a method show\_details().

**CODE 2:**

class Student:

def \_\_init\_\_(self, name, marks):

self.name = name

self.marks = marks

def show\_details(self):

print(f"Name: {self.name}, Marks: {self.marks}")

s1 = Student("Alice", 85)

s2 = Student("Bob", 90)

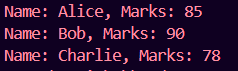
s3 = Student("Charlie", 78)

s1.show\_details()

s2.show\_details()

s3.show\_details()

**OUTPUT 2:**



**QUESTION 3:**

Create a class BankAccount with attributes account\_holder, account\_number, and balance.

● Add methods deposit(amount) and withdraw(amount) that update the balance.

● Add a method show\_balance() that prints the current balance.

● Create an object and perform a deposit, a withdrawal, and show the balance.

**CODE 3**:

class BankAccount:

def \_\_init\_\_(self, account\_holder, account\_number, balance=0):

self.account\_holder = account\_holder

self.account\_number = account\_number

self.balance = balance

def deposit(self, amount):

self.balance += amount

print(f"Deposited {amount}")

def withdraw(self, amount):

if amount <= self.balance:

self.balance -= amount

print(f"Withdrew {amount}")

else:

print("Insufficient balance")

def show\_balance(self):

print(f"Current balance: {self.balance}")

acc1 = BankAccount("David", "12345", 1000)

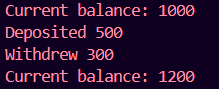
acc1.show\_balance()

acc1.deposit(500)

acc1.withdraw(300)

acc1.show\_balance()

**OUTPUT 3:**



==================================================================

**GitHub:** https://github.com/SushanThakur/2nd-sem-assignment/tree/master/lab-5